Deep Transductive Semi-supervised Maximum Margin Clustering

Gang Chen

January 27, 2015

Abstract

Semi-supervised clustering is an very important topic in machine learning and computer vision. The key challenge of this problem is how to learn a metric, such that the instances sharing the same label are more likely close to each other on the embedded space. However, little attention has been paid to learn better representations when the data lie on non-linear manifold. Fortunately, deep learning has led to great success on feature learning recently. Inspired by the advances of deep learning, we propose a deep transductive semi-supervised maximum margin clustering approach. More specifically, given pairwise constraints, we exploit both labeled and unlabeled data to learn a non-linear mapping under maximum margin framework for clustering analysis. Thus, our model unifies transductive learning, feature learning and maximum margin techniques in the semi-supervised clustering framework. We pretrain the deep network structure with restricted Boltzmann machines (RBMs) layer by layer greedily, and optimize our objective function with gradient descent. By checking the most violated constraints, our approach updates the model parameters through error backpropagation, in which deep features are learned automatically. The experimental results shows that our model is significantly better than the state of the art on semi-supervised clustering.

1 Introduction

In this paper, we investigate the semi-supervised clustering with side information in the form of pairwise constraints. In general, a pairwise constraint between two examples indicates whether they belong to the same cluster or not, which provides the supervision information: a same-label (or must-link) constraint denotes that the pair of instances should be partitioned into the same cluster, while a different-label (or cannot-link) constraint specifies that the pair of instances should be assigned into different clusters [32, 23, 30].

Semi-supervised learning with pairwise constraints, has received considerable attention recently, especially for classification and clustering [32, 3, 11, 23, 6, 30, 34]. On the one hand, it is relatively easy to decide whether two items are similar or not from human in the loop because it often involves little effort from users. On the other hand, the maximum margin techniques have shown promising performance on classification tasks, and thus it has been widely used in semi-supervised clustering [27, 25, 35, 30, 34]. In general, traditional semi-supervised clustering approaches either learn a distance metric based on the pairwise constraints, or leverage discriminative methods, such as k-nearest neighbor (kNN) and support vector machines (SVM) for better clustering performance. However, to collapse examples that belong to the same cluster approximately into a single point cannot always be achieved with simple linear transformations, especially when the data lie on an non-linear manifold. Although kernel methods are widely used for non-linear cases, it is a shallow approach and needs to specify hyper parameters in most situations [27, 30]. Fortunately, recent advances in

the training of deep networks provide a way to learn non-linear transformations of data, which are useful for supervised/unsupervised tasks [8, 2].

Inspired by feature learning [12, 28, 2], we propose a deep transductive semi-supervised clustering approach, which inherits both advantages from deep learning and maximum margin methods. Our method can learn features automatically from observation, kind of learning a metric as in [31]. However, unlike the linear mapping, e.g. Mahalanobis metric [30, 34], our method can learn a non-linear manifold representation, which is helpful for clustering and classification [2]. With the learned features as the input to the semi-supervised maximum margin clustering framework, we can learn the clustering weights. To leverage the unlabeled data, we also incorporate transductive learning to improve the clustering analysis. Through backpropagation, our approach can learn discriminative features via maximum margin techniques. Hence, our model unifies maximum margin, semi-supervised information and deep learning in an joint framework. We pre-train our model with stacked RBMs for feature representations firstly. And then we compute the gradient w.r.t. parameters and optimize our objective function in an alternative manner: data representation and model weights optimization with gradient descent. We test our model over a bunch of data sets and show that it yields accuracy significantly better than the state of the art.

The outline of this paper is as follows. In Section 2, we review the related work. Then, we present the model in Section 3. Section 4 present results of our experiments with the new techniques on a few widely used data sets. Finally we conclude the paper.

2 Related work

The semi-supervised clustering with partial labels generally explores two directions to improve performance: (1) leverage more sophisticated classification models, such as maximum margin techniques [25, 30]; (2) learn a better distance metric [23, 30].

The maximum margin clustering (MMC) aims to find the hyperplanes that can partition the data into different clusters over all possible labels with large margins [33, 26, 35]. Nevertheless, the accuracy of the clustering results by MMC may not be good sometimes due to the nature of its unsupervised learning [14]. Thus, it is interested to incorporate semi-supervised information, e.g. the pairwise constraints, into the recently proposed maximum margin clustering framework. Recent research demonstrates the advantages by leveraging pairwise constraints on the semi-supervised clustering problems [29, 16, 32, 1, 5, 3]. In particular, COPKmeans [11] is a semi-supervised variant of Kmeans, by following the same clustering procedure of Kmeans while avoiding violations of pairwise constraints. MPCKmeans [3] extended Kmeans and utilized both metric learning and pairwise constraints in the clustering process. More recently, [20] show that they can improve classification with pairwise constraints under maximum margin framework. [34] leverage the margin-based approach on the semi-supervised clustering problems, and yield competitive results.

How to learn a good metric over input space is critical for a successful semi-supervised clustering approach. Hence, another direction for clustering is to learn a distance metric [32, 23, 11, 10, 6, 30] which can reflect the underlying relationships between the input instance pairs. The pseudo-metric [23] parameterized by positive semi-definite matrices (PSD) is learned with an online updating rule, that alternates between projections onto PSD and onto half-space constraints imposed by the instance pairs. [32] proposed to learn a distance metric (Mahalanobis) that respects pairwise constraints for clustering. In [6], an information-theoretic approach to learning a Mahalanobis distance function via LogDet divergence is proposed. Recently, a supervised approach to learn Mahalanobis metric is also proposed in [30], by minimizing the pairwise distances between instances in the same cluster, while increasing the separation between data points with dissimilar classes. To handle the data that lies on non-linear manifolds, kernel methods are widely used. Unfortunately, these non-linear embedding algorithms for use is shallow methods.

On the other hand, recent advances in deep learning [12, 28, 2] have sparked great interest in dimension

reduction [13, 31] and classification problems [12, 19]. In a sense, the success of deep learning lies on learned features, which are useful for supervised/unsupervised tasks [8, 2]. For example, the binary hidden units in the discriminative Restricted Boltzmann Machines (RBMs) [18, 9] can model latent features of the data that improve classification. The deep learning for semi-supervised embedding [31] extends shallow semi-supervised learning techniques such as kernel methods with deep neural networks, and yield promising results. The work of [24] is most related to our proposed algorithm. It presented deep learning with support vector machines, which can learn features under discriminative learning framework automatically with labeled data. However, their approach is totally supervised and for classification problems, while our model is for semi-supervised clustering problems. Compared to conventional methods, our model consider both feature learning and transductive principles in our semi-supervised clustering model, so that it can handles complex data distribution and learns a better non-linear mapping to improve clustering performance.

3 Deep Transductive Semi-supervised Maximum Margin Clustering

In this section, we will introduce the transductive semi-supervised maximum margin clustering, with deep features learned simultaneously in an unified framework.

3.1 Overview of our approach

Let $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^N$ ($\mathbf{x}_i \in \mathbb{R}^D$) be a set of N examples, which belongs to K clusters called $\mathcal{Z}$. In addition to the unlabeled data, there is additional partially labeled data in the form of pairwise constraints $C = \{(\mathbf{x}_i, \mathbf{x}_j, \delta(z_i = z_j)\}$, which is a kind of side information to provide whether the two instances $(\mathbf{x}_i, \mathbf{x}_j)$ are from the same cluster or not (indicated by the δ function). Most methods attempt to learn weights $\mathbf{w}^k \in \mathbb{R}^D$, for each cluster $k = [1, K]$, to make these constraints satisfied as much as possible.

Instead of learning a linear mapping or Mahalanobis metric [23, 30], we are interested in a non-linear mapping function. To make it easy to understand, suppose we have learned a nonlinear mapping function $f : \mathbb{R}^D \to \mathbb{R}^d$. Then, for each instance $\mathbf{x} \in \mathcal{X}$, we can get its embedding code $\mathbf{h} = f(\mathbf{x})$ (note that the pairwise constrains also are kept in the coding space). Then given the learned features $\mathbf{h}$, we leverage semi-supervised maximum margin clustering to partition the data. Just like the multi-class classification problems [25], we use the joint feature representation $\Phi(\mathbf{h}, z)$ for each $(\mathbf{h}, z) \in \mathcal{X} \times \mathcal{Z}$

$$\Phi(\mathbf{h}, z) = \begin{bmatrix} \mathbf{h} \cdot \delta(z = 1) \\ \cdots \\ \mathbf{h} \cdot \delta(z = K) \end{bmatrix} \tag{1}$$

where δ is the indicator function (1 if the equation holds, otherwise 0). Correspondently, the hyperplanes for the K clusters can be parameterized by the weight vector $\mathbf{W} \in \mathbb{R}^{(K \times d) \times 1}$, which is the concatenation of weights $\mathbf{w}^k$, for $k = \{1, ..., K\}$. In other words, $\mathbf{W}[(k-1) \times d + 1 : k \times d] = \mathbf{w}^k$. The clustering of testing examples is done in the same manner as the multiclass SVM [25],

$$\max_{z \in [1,K]} \mathbf{W}^T \Phi(f(\mathbf{x}), z) \tag{2}$$

For inference, we first project data into hidden space with function f and then do clustering analysis. The problem left is how to learn the weight parameter $\mathbf{W}$ and the projection function f.

3.2 Objective function

We would like to extend semi-supervised clustering with deep feature learning. Deep learning consists of learning a model with several layers of non-linear mapping. As mentioned before, $\mathbf{h} \in \mathbb{R}^d$ is the mapping code with function f, which is non-linear mappings defined with L-layers neural network, s.t.

$$\mathbf{h}_i = f(\mathbf{x}_i) = \underbrace{f_L \circ f_{L-1} \circ \cdots \circ f_1}_{L \text{ times}}(\mathbf{x}_i) \tag{3}$$

where $\circ$ indicates the function composition, and f_l is logistic function with the weight parameter $\boldsymbol{\theta}_l$ respectively for each layer $l = \{1, .., L\}$, refer further to Sec. 3.3 for more details. With a little abuse of symbols, for any input $\mathbf{x}$, If we denote the output of the l-th layer as $f_{1\to l}(\mathbf{x})$, then we can get $\mathbf{h} = f_{1\to L}(\mathbf{x})$.

In a similar manner as in [20, 34], we will incorporate the pairwise constraint information into the margin-based clustering framework. In addition, we leverage the unlabeled data to separate clusters in large margins, by following transductive learning. Specifically, given the pairwise constraint set $C = \{(\mathbf{x}_i, \mathbf{x}_j, \delta(z_i = z_j))\}$, we first project the dataset $\mathcal{X}$ into embedded space and minimize the following transductive semi-supervised objective function

$$\min_{\mathbf{W},\boldsymbol{\Theta}} \frac{\lambda}{2}||\mathbf{W}||^2 + \frac{1}{n^+}\sum_i \eta_i^+ + \frac{1}{n^-}\sum_j \eta_j^- + \frac{\beta}{UK}\sum_{i\in U}\xi_i \tag{4}$$

$s.t.$

$$\forall s_{i1}, s_{i2} \in \mathcal{Z}, s_{i1} \neq s_{i2}; \text{if } (\mathbf{h}_{i1}, \mathbf{h}_{i2}, \delta(z_{i1}, z_{i2})) \in C^+$$

$$\max_{z_{i1}=z_{i2}} \mathbf{W}^T\Phi(\mathbf{h}_{i1}, \mathbf{h}_{i2}, z_{i1}, z_{i2}) - \mathbf{W}^T\Phi(\mathbf{h}_{i1}, \mathbf{h}_{i2}, s_{i1}, s_{i2}) \geq 1 - \eta_i, \eta_i \geq 0 \tag{5}$$

$$\forall s_{j1}, s_{j2} \in \mathcal{Z}, s_{j1} = s_{j2}; \text{if } (\mathbf{h}_{j1}, \mathbf{h}_{j2}, \delta(z_{j1}, z_{j2})) \in C^-$$

$$\max_{z_{j1}\neq z_{j2}} \mathbf{W}^T\Phi(\mathbf{h}_{j1}, \mathbf{h}_{j2}, z_{j1}, z_{j2}) - \mathbf{W}^T\Phi(\mathbf{h}_{j1}, \mathbf{h}_{j2}, s_{j1}, s_{j2}) \geq 1 - \eta_j, \eta_j \geq 0 \tag{6}$$

$$\forall i \in U, \forall s_i \neq z_i \in \mathcal{Z}$$

$$\max_{z_i} \mathbf{W}^T\Phi(\mathbf{h}_i, z_i) - \mathbf{W}^T\Phi(\mathbf{h}_i, s_i) \geq 1 - \xi_i \tag{7}$$

where $\mathbf{W}$ is the clustering weight in the over the learned feature space, $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_l\}_{l=1}^L$ are the weights for each layer in the deep architecture, and $\mathbf{h}_i$ is the mapping code from $\mathbf{x}_i$ via Eq. 3; $C^+ = \{(\mathbf{h}_i, \mathbf{h}_j, \delta(z_i = z_j))|z_i = z_j\}$ are the same label pairs, with the total number of pairwise constraints $n^+ = |C^+|$, $C^- = \{(\mathbf{h}_i, \mathbf{h}_j, \delta(z_i = z_j))|z_i \neq z_j\}$ are different-label pairs, with $n^- = |C^-|$. U is the number of the unlabeled data (instances), not belong to any pairwise constrains. For convenience, we define $\Phi(\mathbf{h}_i, \mathbf{h}_j, z_i, z_j) = \Phi(\mathbf{h}_i, z_i) + \Phi(\mathbf{h}_j, z_j)$, which means the mapping of a pairwise constraint as the sum of the individual example-label mappings. The multi-layers non-linear mapping function f projects $\mathbf{x}_i$ into $\mathbf{h}_i$, for $i \in [1, N]$. Instead of a linear mapping, the advantage of using a deep network to parametrize the function f is that a multi-layer network is better at learning a non-linear function that is presumably required to collapse classes in the latent space, in particular when the data consists of very complex non-linear structures.

Eqs. 5 and 6 specify the conditions that need to be satisfied, which means that the score for the most possible assigning scheme satisfying the constraints should be greater than that for any other assigning scheme with large margins. More specifically, for any pair $(\mathbf{h}_i, \mathbf{h}_j, 1) \in C^+$, it requires that the largest score for assigning $(\mathbf{h}_i, \mathbf{h}_j)$ into the same cluster should be greater than that for assigning the pair into different clusters by at least 1 (soft margin can be applied here too). Analogously, for any dissimilar pair $(\mathbf{h}_i, \mathbf{h}_j, 0) \in C^-$, the score that they are assigned into the most two different clusters should be greater than that for partitioning them into the same cluster.

Eq. 7 is from the principles of transductive learning, which indicates that the score of the most assigned cluster label is greater at least 1 than that of the runner up from the rest clusters.

The constrained optimization problem in Eq. 4 is hard to solve because the first inequality Eq. 5 and the second inequality Eq. 6 impose all the possible combinations of two clusters for each pairwise constraint. Thus, we transform it into the following equivalent unconstrained function which it is generally easier to solve

$$\begin{aligned}
&\min \frac{\lambda}{2}||\mathbf{W}||^2 \\
&+ \frac{1}{n^+}\Bigg\{1 - \Bigg[\max_{\substack{z_{i1}=z_{i1} \\ (\mathbf{h}_{i1},\mathbf{h}_{i2},1)\in C^+}} \mathbf{W}^T\Phi(\mathbf{h}_{i1},\mathbf{h}_{i2},z_{i1},z_{i2}) - \\
&\qquad \max_{s_{i1}\neq s_{i1}} \mathbf{W}^T\Phi(\mathbf{h}_{i1},\mathbf{h}_{i2},s_{i1},s_{i2})\Bigg]\Bigg\}_+ \qquad (8a) \\
&+ \frac{1}{n^-}\Bigg\{1 - \Bigg[\max_{\substack{z_{j1}\neq z_{j2} \\ (\mathbf{h}_{j1},\mathbf{h}_{j2},0)\in C^-}} \mathbf{W}^T\Phi(\mathbf{h}_{j1},\mathbf{h}_{j2},z_{j1},z_{j2}) - \\
&\qquad \max_{s_{j1}=s_{j2}} \mathbf{W}^T\Phi(\mathbf{h}_{j1},\mathbf{h}_{j2},s_{j1},s_{j2})\Bigg]\Bigg\}_+ \qquad (8b) \\
&+ \frac{\beta}{UK}\sum_{i\in U}\left\{1 - [\max_{z_i}\mathbf{W}^T\Phi(\mathbf{h}_i,z_i) - \max_{s_i\neq z_i}\mathbf{W}^T\Phi(\mathbf{h}_i,s_i)]\right\}_+ \qquad (8c)
\end{aligned}$$

where $\{x\}_+ = max(x,0)$ and $\mathbf{h}_i$ is the projected code of $\mathbf{x}_i$ using Eq. 3. The formula 8a specifies the condition that need to be satisfied for the same label pairwise constrains, while formula 8b denotes the conditions for different-label pairs. The last equation is corresponding to transductive constraints in Eq. 4.

In the objective function, we need to estimate the parameters, the weight $\mathbf{W}$, as well as the weights $\boldsymbol{\theta}_l$ for each layer $l \in [1, L]$ in the deep network. From the objective function, we can compute the gradients w.r.t. $\mathbf{W}$ and $\boldsymbol{\theta}_l$ for $l \in [1, L]$ (via backpropagation) respectively, and gradient-based methods can be used to optimize it. Note that $\mathbf{h}$ (we ignore the subscript for convenience) in the objective function Eq. 8 is the non-linear embedding code from $\mathbf{x}$. Thus, the objective function is not convex anymore, and we can only find a local minimum. In practice, we find L-BFGS cannot work well and easily trap into a bad local minimum. In our work, we use (sub)gradient descent to optimize the objective function, by projecting the training data with f and optimizing the objective function in an alternative manner.

3.3 Parameter learning

We learn the parameters in an alternative manner: (1) data projection, given the model parameters; (2) and then optimize model parameters with gradient descent. To compute the gradients of the parameters, we need to find the most violated constraints first. For the same label pairs, we have the following most violated set:

$$A^+ = \left\{(\mathbf{h}_i,\mathbf{h}_j,\delta(z_{i1}=z_{i2}))\in C^+ | \max_{z_{i1}=z_{i1}} \mathbf{W}^T\Phi(\mathbf{h}_{i1},\mathbf{h}_{i2},z_{i1},z_{i2}) - \max_{s_{i1}\neq s_{i1}} \mathbf{W}^T\Phi(\mathbf{h}_{i1},\mathbf{h}_{i2},s_{i1},s_{i2}) < 1\right\} \quad (9)$$

For the different-label pairs, we denote the most violated set as

$$A^- = \left\{(\mathbf{h}_i,\mathbf{h}_j,\delta(z_{j1}=z_{j2}))\in C^- | \max_{z_{j1}\neq z_{j2}} \mathbf{W}^T\Phi(\mathbf{h}_{j1},\mathbf{h}_{j2},z_{j1},z_{j2}) - \max_{s_{j1}=s_{j2}} \mathbf{W}^T\Phi(\mathbf{h}_{j1},\mathbf{h}_{j2},s_{j1},s_{j2}) < 1\right\} \quad (10)$$

Then, we compute the gradient w.r.t. $\mathbf{W}$

$$
\begin{aligned}
d\mathbf{W} &= \lambda \mathbf{W} + \\
&- \frac{1}{n^+} \sum_{(\mathbf{h}_{i1}, \mathbf{h}_{i2}, 1) \in A^+} \Big[\Phi(\mathbf{h}_{i1}, \mathbf{h}_{i2}, z_{i1}^+, z_{i2}^+) - \Phi(\mathbf{h}_{i1}, \mathbf{h}_{i2}, z_{i1}^-, z_{i2}^-) \Big] \\
&- \frac{1}{n^-} \sum_{(\mathbf{h}_{j1}, \mathbf{h}_{j2}, 0) \in A^-} \Big[\Phi(\mathbf{h}_{j1}, \mathbf{h}_{j2}, z_{j1}^-, z_{j2}^-) - \Phi(\mathbf{h}_{j1}, \mathbf{h}_{j2}, z_{j1}^+, z_{j2}^+) \Big] \\
&- \sum_{i \in U} \frac{\beta}{UK} \Big[\Phi(\mathbf{h}_i, z_i^+) - \Phi(\mathbf{h}_i, s_i^+) \Big],
\end{aligned} \tag{11}
$$

where $(z_{i1}^+, z_{i2}^+) = \max_{z_{i1}=z_{i2}} \mathbf{W}^T \Phi(\mathbf{h}_{i1}, \mathbf{h}_{i2}, z_{i1}, z_{i2})$, $(z_{i1}^-, z_{i2}^-) = \max_{z_{i1} \neq z_{i1}} \mathbf{W}^T \Phi(\mathbf{h}_{i1}, \mathbf{h}_{i2}, z_{i1}, z_{i2})$; and for the unlabeled set $z_i^+ = \max_{z_i} \Phi(\mathbf{h}_i, z_i)$ and $s_i^+ = \max_{s_i \neq z_i^+} \Phi(\mathbf{h}_i, s_i)$

In order to learn discriminative features, we also need to estimate the weights in the multi-layer network. Note that for each pair $(\mathbf{h}_i, \mathbf{h}_j)$, if it violates the constraints in Eqs. 5 and 6, then we can compute the gradient w.r.t. $\mathbf{h}_i$ and $\mathbf{h}_j$ respectively, which will be used to calculate the gradients of $\boldsymbol{\theta}_l$ for $l \in [1, L]$ in the deep network. We use $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_N]$ as the concatenation of all the hidden codes, where $\mathbf{H} \in \mathcal{R}^{d \times N}$, with each column $\mathbf{H}(:, i) = \mathbf{h}_i$.

For the positive pairs, we have

$$
d\mathbf{h}_{i1} = -\frac{1}{n^+} \sum_{i1} [\mathbf{W}_{z_{i1}^+} - \mathbf{W}_{z_{i1}^-}] \tag{12a}
$$

$$
d\mathbf{h}_{i2} = -\frac{1}{n^+} \sum_{i2} [\mathbf{W}_{z_{i1}^+} - \mathbf{W}_{z_{i2}^-}] \tag{12b}
$$

where $\mathbf{W}_{z_{i1}^+}$ indicates the weight vector corresponding to the cluster label z_{i1}^+ in the whole weight matrix $\mathbf{W}$. More specifically, $\mathbf{W}_{z_{i1}^+} = \mathbf{W}[(z_{i1}^+ - 1) \times d + 1 : z_{i1}^+ \times d]$

For the negative pairs, we can get

$$
d\mathbf{h}_{j1} = -\frac{1}{n^-} \sum_{j1} [\mathbf{W}_{z_{j1}^-} - \mathbf{W}_{z_{j1}^+}] \tag{13a}
$$

$$
d\mathbf{h}_{j2} = -\frac{1}{n^-} \sum_{j2} [\mathbf{W}_{z_{j2}^-} - \mathbf{W}_{z_{j1}^+}] \tag{13b}
$$

For the unlabeled instances, we have

$$
d\mathbf{h}_i = -\frac{\beta}{UK} \sum_{i} [\mathbf{W}_{z_i^+} - \mathbf{W}_{s_i^+}] \tag{14}
$$

Given the gradient of $d\mathbf{h}_i$ for each hidden code, we can get the gradient w.r.t. $\mathbf{H}$ as

$$
d\mathbf{H}(:, i) = d\mathbf{h}_i \tag{15}
$$

where $d\mathbf{h}_i$ can be calculated according to Eqs. 12 and 13. Then, we can calculate the gradients w.r.t. lower level weights with back-propagation. For example $d\boldsymbol{\theta}_L = d\mathbf{H} \times \big(f_{1 \to L}(\mathcal{X}) \cdot (1 - f_{1 \to L}(\mathcal{X}))\big)$, where $\times$ representas matrix multiplication, and $\cdot$ indicates pointwise product.

Initialization: We used stacked RBMs to initialize the weights layer by layer greedily in the deep network, with contrastive divergence [12] (we used CD-1 in our experiments). Note that we used gaussian RBMs for the continuous data in the first layer, otherwise we used binary RBMs.

Algorithm 1

Input: the training data $\mathcal{X}$, pairwise constraints C, the number of clusters K, the number of iterations T, λ, and β;
Initialize $\mathbf{W}$;
Initialize $\mathbf{w}_l$ for $l = \{1, ..., L\}$ layer-by-layer greedily;
for $i = 1; i <= T; i++$ **do**
  if the objective in Eq. 8 has no significant changes, break;
  project all training data $\mathcal{X}$ into latent space via Eq. 3;
  find the most violated constrains according to Eqs. 9 and 10
  compute the gradient w.r.t. $\mathbf{W}$ via Eq. 11;
  compute the gradient w.r.t. $\mathbf{H}$ via Eq. 15;
  compute the gradient w.r.t. $\mathbf{\Theta} = \{\boldsymbol{\theta}_l\}_{l=1}^{L}$ with backpropagation;
  update the parameters with gradient descent via Eq. 16;
end for
Return model parameters $\mathbf{W}$ and $\{\mathbf{w}_l\}_{l=1}^{L}$, as well as average accuracy;

In our deep model, the weights from the layers 1 to L are $\boldsymbol{\theta}_l$ respectively, for $l = \{1, .., L\}$, and the top layer L has weight $\boldsymbol{\theta}_L$. We first pre-train the L-layer deep structure with RBMs layer by layer greedily. Thus, our deep network can learn parametric nonlinear mapping from input $\mathbf{x}$ to output $\mathbf{h}$, $f : \mathbf{x} \rightarrow \mathbf{h}$. Specifically, we think RBM is a 1-layer deep network, with weight $\boldsymbol{\theta}_1$. For example, for 1-layer DBN, we have $\mathbf{h} = f_1(\mathbf{x}) = \text{logistic}(\boldsymbol{\theta}_1^T[\mathbf{x}, 1])$, where we extend $\mathbf{x} \in \mathbb{R}^D$ into $[\mathbf{x}, 1] \in \mathbb{R}^{(D+1)}$ in order to handle bias in the non-linear mapping. Given the output of the current layer as the input to the next layer, we can learn each layer weight greedily.

As for the clustering weight $\mathbf{W}$, we take a similar strategy as in [34] to initialize it.

Parameter updating: In our model, we use the gradient descent to update the model parameters. We also tried L-BFGS [4, 22] to update model parameters, but it did not perform well. In our model, we can update the model parameters as follows,

$$\begin{aligned} &\mathbf{W} \leftarrow \mathbf{W} - \gamma_{\mathbf{W}} d\mathbf{W}, \\ &\boldsymbol{\theta}_l \leftarrow \boldsymbol{\theta}_l - \gamma_{\boldsymbol{\theta}_l} d\boldsymbol{\theta}_l, l \in \{1, ..., L\} \end{aligned} \tag{16}$$

where $\gamma_{\mathbf{W}}$ is the learning rate for the clustering weight $\mathbf{W}$, and $\gamma_{\boldsymbol{\theta}_l}$ is the learning rate for weights $\boldsymbol{\theta}_l$ in the deep neural network. Thus, our method alternates between data projection and parameter optimization. For more details, refer to algorithm 1.

After we learned the model parameters, we can do cluster analysis according to Eq. 2.

4 Experiments

In this section, we presented a set of experiments comparing our method to the state of the art semi-supervised clustering methods on a wide range of data sets, including UCI data sets and the Reuters dataset in Table 1, as well as the MNIST digits, COIL-20 and COIL-100 datasets. We also evaluated whether the transductive constraint in Eq. (7) is helpful or not in the clustering analysis.

4.1 Experimental setup

In the experiments, we compared our method to the state of the art semi-supervised clustering approaches, including Xing [32], ITML [6], KISSME [17] and CMMC [34]. Note that Xing, ITML and KISSME are the

Dataset	Description		
	#vectors	dim	#clusters
Glass	214	9	7
Wdbc	569	32	2
Wine	178	13	3
Sonar	208	60	2
Image Segmentation	2310	19	7
Reuters	8293	18933	65

Table 1: Descriptions of the UCI datasets and the Reuters dataset.

Methods	Accuracy (%)						Adjusted Rand Index					
	Glass	Wdbc	Wine	Sonar	Segmentation	Reuters	Glass	Wdbc	Wine	Sonar	Segmentation	Reuters
Xing [32]	46.2	91.9	81.5	53.4	28.5	44.3	0.214	0.70	0.584	0.02	0.12	0.14
ITML [6]	47.4	**92.1**	70.2	69.2	30.0	45.0	**0.223**	**0.71**	0.520	0.14	0.14	0.15
KISSME [17]	36.5	77.9	65.2	67.3	27.6	49.7	0.07	0.287	0.466	0.12	0.09	0.17
CMMC [34]	43.2	89.5	97.1	72.1	51.4	66.5	0.217	0.620	0.918	0.191	0.35	0.22
Our method	**50.9**	91.5	**98.8**	**72.6**	**57.1**	**72.7**	0.219	0.689	**0.965**	**0.20**	**0.41**	**0.56**

Table 2: The experimental comparison on the UCI data sets and the Reuters data set. For the real UCI datasets, our method outperforms other methods significantly, except on the Glass and Wdbc data sets. Our method is remarkably better on the Reuters dataset, with both accuracy and Rand index.

semi-supervised approaches for metric learning (Mahalanobis). Thus, we used those methods to learn the metric and calculate the distances between all instances, then we used the kernel k-means [7] for clustering. Our method and CMMC are similar, which can be directly optimized for clustering.

As for parameter setting, we set $\lambda = 0.02$ and $\beta = 1$. The learning rate $\gamma_{\mathbf{W}}$ decreases in the iterations in our model, by setting $\gamma_{\mathbf{W}} = \frac{1}{\lambda \times (i+1)}$, where i is the index for iterations; while the learning rate for weighs in the deep network fixed, with $\gamma_{\boldsymbol{\theta}_l} = 0.01$, for $l = \{1, ..., L\}$. Without other specification, our model used the one hidden layer with 100 units on most data sets, except on the MNIST and UCI data sets.

We tested our method on two tasks: pairwise classification and clustering analysis. As for pairwise classification, we randomly sampled 200 pairs of constraints (around 100 must-links and 100 cannot links), of which we used 100 pairwise constraints (50 must-links and 50 cannot links) as the training set, and the rest 100 pairs as the testing sets. Then we used the receiver operating characteristic (ROC) to evaluate the performance.

As for the clustering analysis, we used the pairwise constraints sampled to train our model, then we use the learned model for clustering analysis. We used the accuracy (the most possible matching between the obtained labels and the original true labels, refer to [34]) and adjusted Rand Index [15, 21] to evaluate our method in all the experiments.

4.2 Results

UCI data sets: In the experiment, we selected the five widely used data sets from the UCI machine learning repository[1], which has different dimension and categories, shown in Table 1. As for the number of hidden units in our model, we set the number of hidden nodes to be 100 on the sonar data set and 64 on the other UCI data sets. For the each data set, we randomly sampled 200 pairwise constraints, of which 100 pairs were used for training and the rest to test the pairwise classification performance. While for clustering performance, we test the model on all the data elements. We compared our method to the state of the art methods, and clustering results are shown in Table. (2). It demonstrates that our method outperforms other

[1] `https://archive.ics.uci.edu/ml/datasets.html`

methods on almost all the data sets, especially for the data with the larger number of classes. We also show the performance of our method on the pairwise classification task in Fig. 1. Except on the Wdbc and glass data sets, our method yields completive and even better results than other methods.

Reuters data set: We used the Reuters21578[2], which has the total 8293 documents with 18933 dimensional features for each document, belonging to 65 categories. Because the Reuters data set has high dimension, we first projected it into 400 dimensions with PCA. Then we set the number of hidden nodes to be 100 in our model. The clustering performance is shown in Table. (2). It demonstrates that our method is significantly better than other methods. Again, our method yields remarkably better pairwise classification result, shown in the right bottom of Fig. 1.

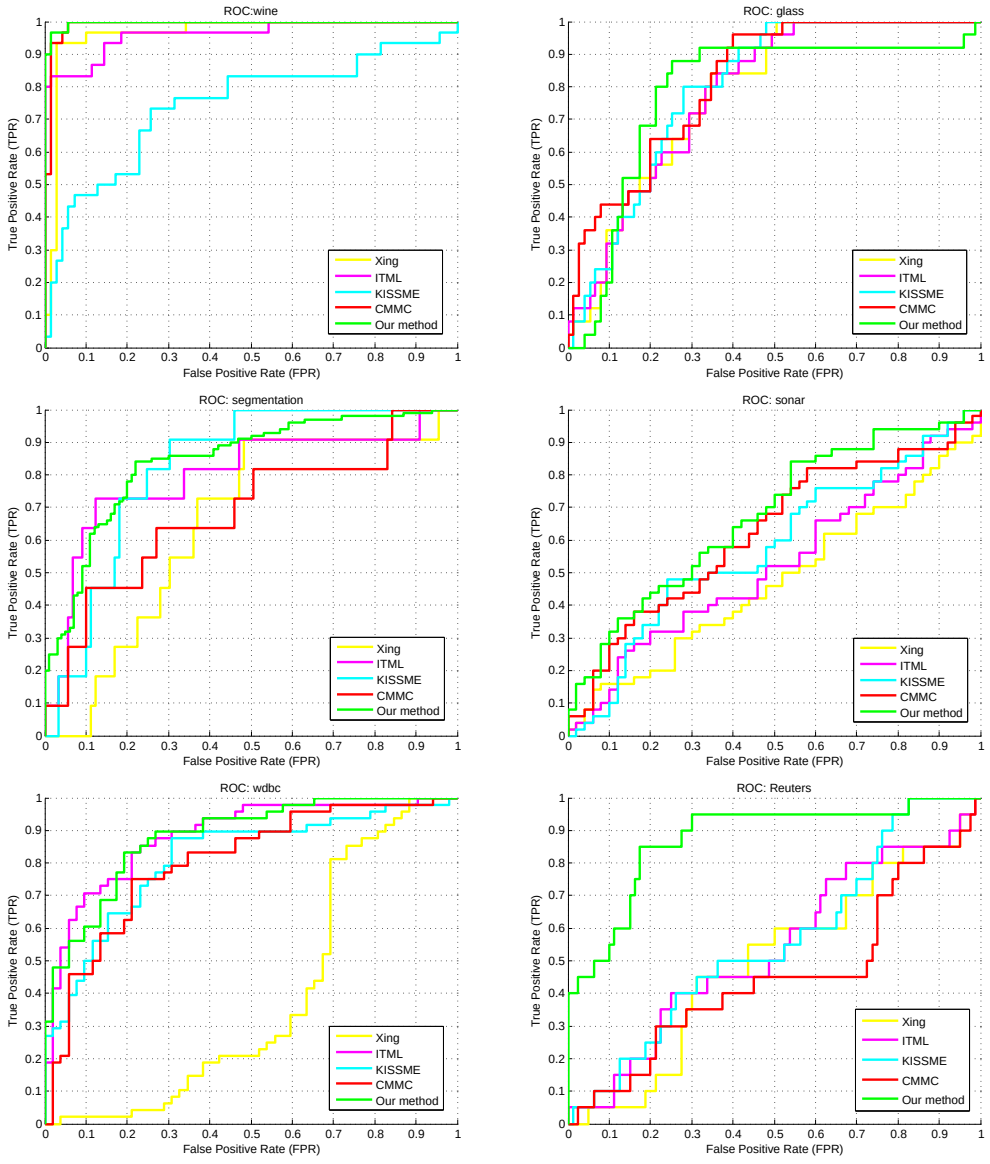


Figure 1: The pairwise classification results on the five UCI data sets and the Reuters data set (the right bottom).

[2]http://www.cad.zju.edu.cn/home/dengcai/Data/TextData.html

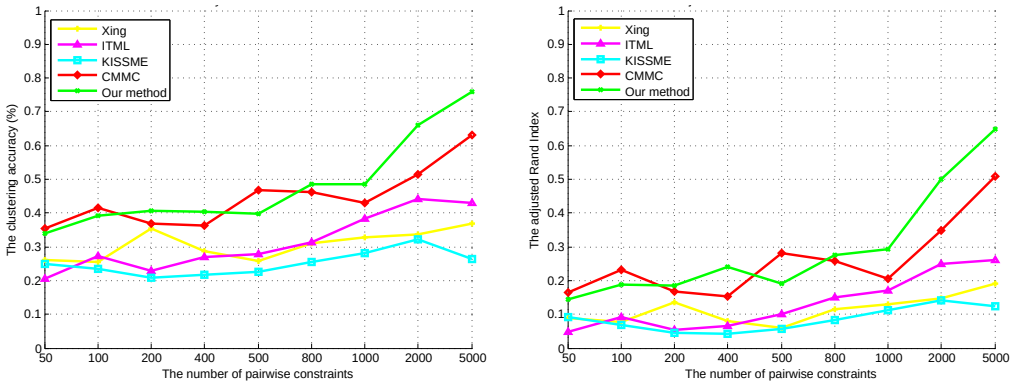


Figure 2: The clustering performance comparison on the MNIST digits by varying the number of training pairs, evaluated with accuracy and rand index respectively. It demonstrates that our method is significantly better than other methods for clustering analysis.

MNIST dataset: The MNIST digits[3] consists of 28×28-size images of handwriting digits from 0 through 9 with a training set of 60,000 examples and a test set of 10,000 examples, and has been widely used to test character recognition methods. In the experiment, we randomly sampled a subset with 5000 images from the training sets to test our method and other baselines. In the experiment, we use a three-layer deep structure for MNIST digits, with hidden nodes [400 200 100] respectively on each layer. We tested how the clustering performance changes when the number of pairwise constraints varies. The experimental comparisons between our method and other baselines are shown in Fig. 2. It demonstrates that the clustering accuracy is increasing with more pairwise constraints. And it also shows our method is better than other baselines in most cases when varying the number of training pairs.

To evaluate whether the transductive constraint in our model in Eq. 4 is helpful or not for clustering, we set $\beta = 0$ to get rid of the transductive condition in Eq. 7, and the experimental results are shown in Fig. 3. We argue that the result in Fig. 3 is consistent with common sense. The smaller the number of pairwise constraints, the higher uncertainty when we do inference. Thus, transductive learning has no advantage when the number of constraints is small. But it performs better with more constraints in Fig. 3. When more and more pairwise constraints are available, there's no need to incorporate transductive principles in the model. To sum up, it demonstrates that the transductive constraint in our model is remarkably helpful for the semi-supervised clustering analysis.

COIL data set: We test our method on both COIL-20 and COIL-100 image data sets. The COIL-20 data set[4] has total 1440 images, with size 128×128. It is divided into 20 classes of objects, with 72 images for each object. In our experiments, we used the processed version, which contains images for all of the objects in which the background has been discarded, and furthermore we resized all the images into 32×32 for space and time concern. The COIL-100 data set consists of 7200 images, partitioned into 100 classes. Similarly, we also resized the images into 32×32 before learning clustering model.

The clustering performance is shown in Fig. 5, and it demonstrates that our method is better than other methods with both stability and clustering accuracy. Where's the performance gain from in Fig. 5? deep learning or transductive learning? To answer this question, we evaluated whether transductive training is helpful when the number of pairwise constraints is limited. In Figs. 4(a) and (b), it shows the results on 20 classes, and demonstrates that transductive learning is helpful when the number of training pairs is in range [400 2000]. But with more and more training constraints, transductive learning cannot improve the performance too much. In Figs. 4(c) and (d), it shows the results on COIL with 100 classes, and indicates

[3] `http://yann.lecun.com/exdb/mnist/`

[4] `http://www.cs.columbia.edu/CAVE/software/softlib/coil-20.php`

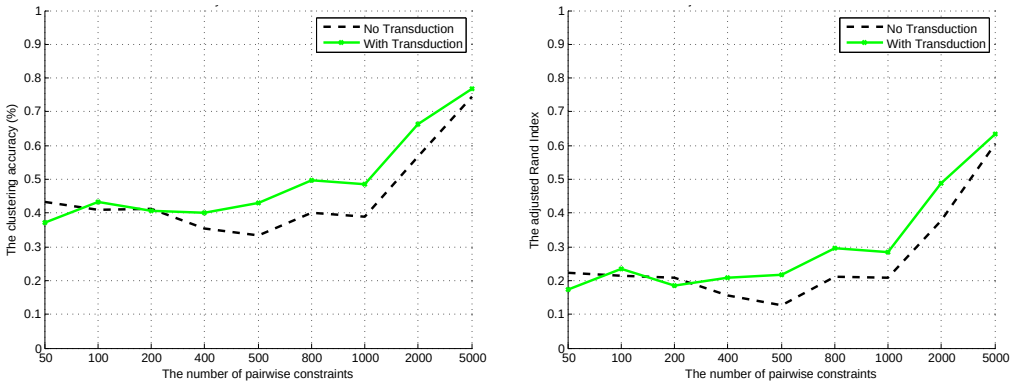
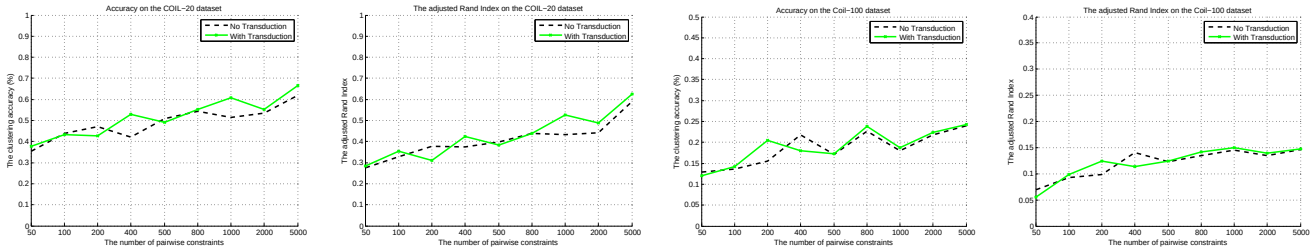


Figure 3: The comparison between with and without transductive principles for our method. (a) and (b) show the results (evaluated with accuracy and rand index respectively) on the MNIST data set.

that transductive learning cannot improve the performance much when the number of classes is large. We think the reason that transductive learning cannot perform well in Figs. 4(c) and (d) is that it cannot infer label well with large margin on the dataset with a larger number of clusters. We argue when we have more data and more clusters, it is harder to partition the data well, and more difficult to find a better hyperplane to separate one cluster well from the others with large margin. In other words, it is harder to satisfy the condition in Eq. 7. Compared to the COIL dataset, transductive learning yields a larger gain on the MNIST data set in Fig. 3. Thus, transductive learning is helpful when the number of classes is small and the data is well distributed (compact within the same cluster, and separated between different clusters).

Thus, the most performance gain on the COIL-100 data set in Fig. 5 is from deep learning, according to the above analysis.



(a) (b) (c) (d)

Figure 4: The comparison between with and without transductive principles for our method. (a) and (b) show the results (evaluated with accuracy and rand index respectively) on the COIL-20 data set; (c) and (d) are the results on the COIL-100 dataset, with accuracy and rand index respectively. For the 20 classes, it shows that the transductive learning is helpful when the number of training pairs is small. However, for the 100 classes, the transductive learning cannot improve the performance. It demonstrates that it is better to leverage transductive principles when the number of classes is relative smaller.

5 Conclusions

In this paper, we propose a deep transductive semi-supervised maximum margin clustering approach. One the one hand, we leverage deep learning to learn non-linear representations, which can be used as the input to the semi-supervised clustering model. On the other hand, we incorporate the non-label instances into

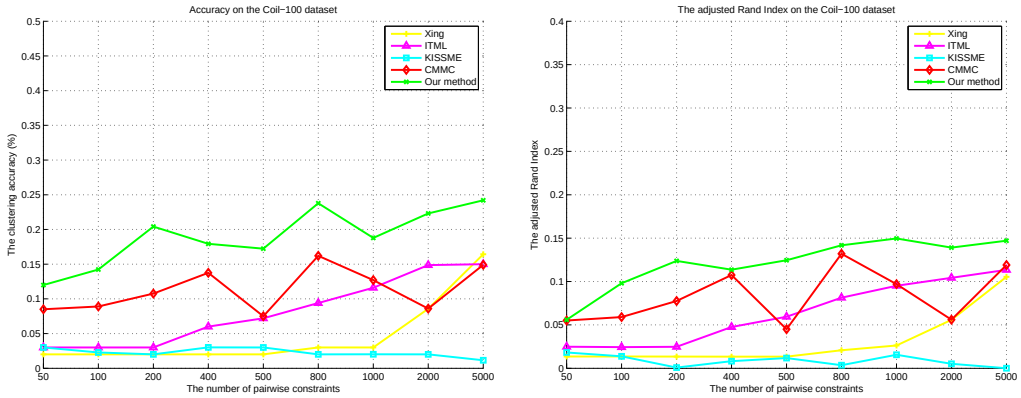


Figure 5: The clustering comparison on the COIL-100 data set by varying the number of training pairs. It shows that our method outperforms other methods significantly.

our semi-supervised clustering framework. Thus, our model unifies transductive learning, deep learning, maximum margin and semi-supervised clustering in one framework. Compared to conventional methods, our approach can learn non-linear mappings as well as leveraging transductive information to improve clustering performance. We pretrain the deep structure with stacked restricted Boltzmann machines layer by layer greedily for feature representations and optimize our objective function with gradient decent. We demonstrate the advantages of our model over the state of the art in the experiments.

References

[1] A. Bar-Hillel, T. Hertz, N. Shental, and D. Weinshall. Learning distance functions using equivalence relations. In *In Proceedings of the Twentieth International Conference on Machine Learning*, pages 11–18, 2003.

[2] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *PAMI*, pages 1798–1828, 2013.

[3] M. Bilenko, S. Basu, and R. J. Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 11–, New York, NY, USA, 2004. ACM.

[4] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput.*, 16(5):1190–1208, Sept. 1995.

[5] D. Cohn, R. Caruana, and A. Mccallum. Semi-supervised clustering with user feedback. Technical report, 2003.

[6] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 209–216, New York, NY, USA, 2007. ACM.

[7] I. S. Dhillon, Y. Guan, and B. Kulis. Kernel k-means, spectral clustering and normalized cuts. In *KDD*, 2004.

[8] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio. Why does unsupervised pre-training help deep learning? *J. Mach. Learn. Res.*, 11:625–660, Mar. 2010.

[9] A. Gelfand, Y. Chen, L. van der Maaten, and M. Welling. On herding and the perceptron cycling theorem. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 694–702, 2010.

[10] A. Globerson and S. T. Roweis. Metric learning by collapsing classes. In *NIPS*, 2005.

[11] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In *Advances in Neural Information Processing Systems 17*, pages 513–520. MIT Press, 2004.

[12] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural Comput.*, 18(7):1527–1554, jul 2006.

[13] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, July 2006.

[14] Y. Hu, J. Wang, N. Yu, and X.-S. Hua. Maximum margin clustering with pairwise constraints. In *ICDM*, pages 253–262. IEEE Computer Society, 2008.

[15] L. Hubert and P. Arabie. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985.

[16] D. Klein, S. D. Kamvar, and C. D. Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *Proceedings of the Nineteenth International Conference on Machine Learning*, ICML '02, pages 307–314, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[17] M. Koestinger, M. Hirzer, P. Wohlhart, P. M. Roth, and H. Bischof. Large scale metric learning from equivalence constraints. In *Proc. IEEE Intern. Conf. on Computer Vision and Pattern Recognition*, 2012.

[18] H. Larochelle and Y. Bengio. Classification using discriminative restricted boltzmann machines. In *ICML*, pages 536–543, New York, NY, USA, 2008. ACM.

[19] H. Larochelle, M. Mandel, R. Pascanu, and Y. Bengio. Learning algorithms for the classification restricted boltzmann machine. *J. Mach. Learn. Res.*, 13(1):643–669, Mar. 2012.

[20] N. Nguyen and R. Caruana. Improving classification with pairwise constraints: A margin-based approach. In *ECML/PKDD (2)*, pages 113–124, 2008.

[21] W. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[22] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[23] S. Shalev-Shwartz, Y. Singer, and A. Y. Ng. Online and batch learning of pseudo-metrics. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 94–, New York, NY, USA, 2004. ACM.

[24] Y. Tang. Deep learning using support vector machines. In *Workshop on Representational Learning, ICML 2013*, volume abs/1306.0239, 2013.

[25] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *JMLR*, pages 1453–1484, 2005.

[26] H. Valizadegan and R. Jin. Generalized maximum margin clustering and unsupervised kernel learning. In B. Schlkopf, J. Platt, and T. Hoffman, editors, *NIPS*, pages 1417–1424. MIT Press, 2006.

[27] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., 1995.

[28] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *J. Mach. Learn. Res.*, 11:3371–3408, Dec. 2010.

[29] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl. Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 577–584, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[30] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.*, 10:207–244, June 2009.

[31] J. Weston and F. Ratle. Deep learning via semi-supervised embedding. In *International Conference on Machine Learning*, 2008.

[32] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS 15*, pages 505–512. MIT Press, 2003.

[33] L. Xu, J. Neufeld, B. Larson, and D. Schuurmans. Maximum margin clustering. In *NIPS17*, pages 1537–1544, 2005.

[34] H. Zeng and Y. ming Cheung. Semi-supervised maximum margin clustering with pairwise constraints. *IEEE Trans. Knowl. Data Eng.*, 24(5):926–939, 2012.

[35] J. Zhang and R. Yan. On the value of pairwise constraints in classification and consistency. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 1111–1118, New York, NY, USA, 2007. ACM.